Autonomicity – An Antidote for Complexity?

Roy Sterritt
School of Computing and Mathematics
Faculty of Engineering
University of Ulster
Northern Ireland
r.sterritt@ulster.ac.uk

Mike Hinchey
NASA Goddard Space Flight Center
Software Engineering Laboratory
Greenbelt, MD 20771
USA
michael.g.hinchey@nasa.gov

Abstract

Autonomic Computing and other self-managing system initiatives are emerging as significant new vision for the design and development of complex computer systems. The purpose of this paper is to consider how Autonomous and Autonomic Systems can provide a framework for tackling complexity and overcoming the problems of its (unavoidable) inherent existence in certain classes of systems.

1. Introduction

Computer systems are becoming increasingly demanding, challenging and complex. Along with the "internet revolution" we have seen the convergence of the traditional telecommunication industry and the data communications sector of the computer industry. In the future, it is likely that this will be seen as only the beginning of the merging of these large industries. While this offers the potential for each industry to build on the techniques and advances of the other, it also guarantees an ever increasingly complex world. Realization of future visions of computing, whether it is invisible, world, ubiquitous, pervasive, utility, grid, ambient intelligence, semantic web, etc., will require a coming to terms with complexity.

Organizations that research and develop complex computer systems are facing additional market conditions such as demands for more functionality, ever decreasing time-to-market, domain-expert shortage and high employment costs, leading to a requirement to utilize only semi-skilled employees.

This results in the need for self-managing systems and new development approaches that can deal with real-life complexity and uncertainty. The challenge is to produce practical methodologies and techniques for the development of such self-managing systems, so that they may be leveraged to deal with complexity.

2. The Problem is Complexity

The world is becoming an ever-increasing complex place. In terms of computer systems this complexity has been confounded by the drive towards cheaper, faster and smaller hardware and functionally rich software. The infiltration of the computer into every day life has made the reliance on them critical. As such, there is an increasing need throughout design, development and operation of computer systems to cope with this complexity and the inherent uncertainty within. There is an increasing need to change the way we view computing; there is a need to realign towards facing up to computing in a complex world.

The IT industry is a marked success; within a 50 year period it has grown to become a trillion dollar per year industry obliterating barriers and setting records with astonishing regularity [1][2]. Throughout this time the industry has had a single focus, namely to improve performance [3] which has resulted in some breathtaking statistics [4]:

- Performance/price ratio doubles around every 18 months,
- resulting in 100 fold per decade;
- Progress in the next 18 months will equal ALL previous progress;
- New storage = sum of all old storage, ever;
- New processing = sum of all old processing;
- Aggregate bandwidth doubles in 8 months.

This performance focus has resulted in the emergence of a small number of critical inherent behaviors in the way the industry operates when designing, developing and deploying hardware, software, and systems [3]:

- That humans can achieve perfection; that they avoid making mistakes during installation, upgrade, maintenance or repair.
- Software will eventually be bug free; the focus of companies has been to hire better programmers, and universities to train better software engineers in development life-cycle models.
- Hardware mean-time between failure (MTBF) is already very large -approximately 100 years- and will continue to increase.
- Maintenance costs are irrelevant compared to purchase price. That maintenance is a function of price as such cheaper in the first place helps keep maintenance costs lower.

When made explicit is this way, it is obvious that these implicit behaviors are flawed and result in contributing factors to the complexity problem.

Within the last decade, problems have started to become more apparent. For an industry that is used to metrics always rising saw some key decreases, Figure 1 [1] highlights that key modern day systems – cell phones and the internet, have seen a decline in availability, changing the established trend of their counterparts.

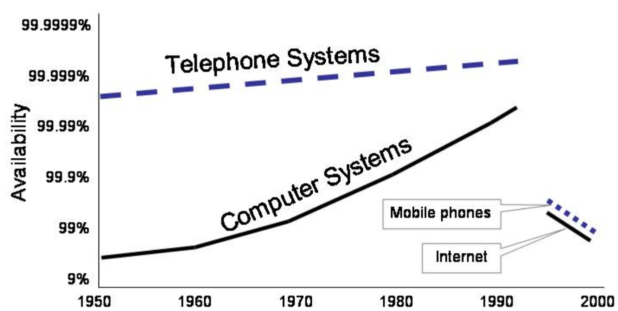

Figure 1 Systems Availability over the Decades [1]

When you also consider how dependant we have become on our systems and how much it costs for a single hour of downtime; for instance, in 2000: $6.5m brokerage operations, $2.5m credit card authorization and $¼m for ebay [3][5], highlights how much we increasingly require our complex systems to be dependable.

3. Dependability through Autonomicity

3.1 Dependability

It may be viewed that Dependability has come of age with the recent inaugural issue of IEEE Transactions on Dependable and Secure Computing. Dependability is defined as that property of a computer-based system that enables reliance to be placed on the service it delivers. That service is its behavior as perceived by other systems or its human users [6].

Figure 2 [6]-[9] (which has been updated in [15]) depicts the concepts of dependability in terms of threats to, attributes of, and the means by which, dependability is attained.

The effectiveness of these four mechanisms has a substantial influence on the dependability of a computer-based system.

Randell describes dependability in terms of *failures*, *faults* and *errors*, arguing that they follow a “fundamental chain” [6], thus:

… ➔failure ➔ fault ➔ error ➔ failure ➔ fault ➔…

More abstractly, this can be described by the sequence:

… ➔event ➔ cause ➔ state ➔ event ➔ cause ➔…

For example, the failure of a system (event) occurs when a fault is encountered during its operation (cause), because of an error in its implementation (state). This might be attributed to a failure in the test process (event) because the relevant code was not exercised (cause) meaning that the test suite was incomplete (state).

These chains may of course be broken at any stage in the chain by effective fault means (fault prevention, fault tolerance, fault removal and fault forecasting - as in Figure 2).

Overall, the breadth of issues involved suggests the need for a holistic approach to designing dependable systems.[36]

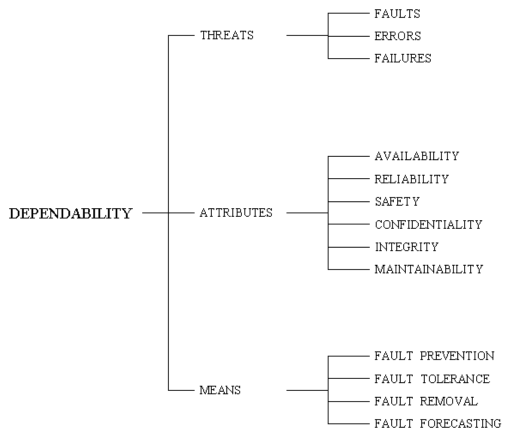

Figure 2 The Dependability Tree

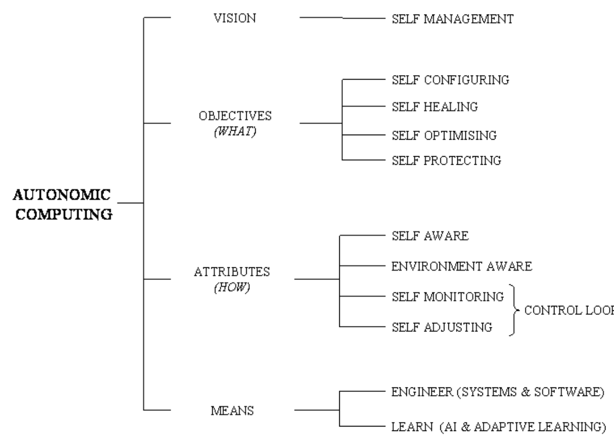

Figure 3 Autonomic Computing Tree

3.2 Autonomic Computing

Autonomic Computing, launched by IBM in 2001 [2], [10]-[13], is emerging as a significant new strategic and holistic approach to the design of computing systems. Two of IBM's main objectives are to reduce the total cost of ownership of systems and to find better ways of managing their increasing complexity.

As well as IBM, many major software and system vendors, such as HP, Sun, Cisco and Microsoft have established strategic initiatives to help create computer systems that manage themselves, concluding that this is the only viable long-term solution.

As the name implies, the influence for the new paradigm is the human body's autonomic system, which regulates vital bodily functions such as the control of heart rate, the body's temperature and blood flow—all without conscious effort.

The desire for automation and effective robust systems is not new; in fact this may be considered an aspect of systems and software engineering best practice. Similarly, the desires for systems self-awareness, awareness of the external environment, and the ability to adapt, are also not new, being major goals of artificial intelligence (AI) research for many years. What may be considered new in Autonomic Computing is its overall breadth of vision and scope.

Research in Autonomic Computing is likely to see a greater collaboration between the AI and software engineering fields. Such collaboration has been motivated by increasing system complexity and a more demanding user community. For example, software engineers have used AI techniques to provide more sophisticated support for user interfaces and to help address soft issues in the development and operation of software. Likewise, the AI community has increasingly been looking to software engineering for disciplined methodologies to support the production of intelligent systems.

Consequently, Autonomic Computing is perhaps best considered a strategic refocus for the engineering of effective systems rather than a revolutionary new approach [14].

The overall goal of Autonomic Computing is the creation of self-managing systems; these are proactive, robust, adaptable and easy to use. Such objectives are achieved though *self-protecting*, *self-configuring*, *self-healing* and *self-optimizing* activities, as indicated in Figure 3 [36].

To achieve these objectives a system must be both *self-aware* and *environment-aware*, meaning that it must have some concept of the current state of both itself and its operating environment. It must then *self-monitor* to recognize any change in that state that may require modification (*self-adjusting*) to meet its overall self-managing goal. In more detail, this means a system having knowledge of its available resources, its components, their desired performance characteristics, their current status, and the status of inter-connections with other systems. This self-monitoring and self-adjusting forms a feedback control loop between the managed component and the autonomic manager.

The ability to operate in a heterogeneous environment requires the use of open standards to understand and communicate with other systems.

In effect, autonomic systems are proactive in their operation, hiding away much of the associated complexity from users.

Self-healing is concerned with ensuring effective recovery under fault conditions, without loss of data or noticeable delays in processing, while identifying the fault and repairing it, where possible. Fault prediction techniques might also be used, leading to re-configuration to avoid the faults concerned or reduce the likelihood of their re-occurrence.

With self-optimization, the system seeks to optimize its operations in both proactive and reactive ways.

With self-protection, a system will defend itself from malicious attack and may also have to self-heal when problems are detected, or self-optimize to improve protection.

With self-configuring, the system may automatically install, configure and integrate new software components seamlessly to meet defined business strategies.

IBM discusses the characteristics or 'elements' of Autonomic Computing in more detail in its manifesto [10]. This is being expanded throughout the research community as witnessed by the uptake of workshops and conferences on the topic such as [22]-[27].

3.3 Autonomic Computing and Dependability

Randell and colleagues [7]-[9] give two main reasons for their interest in and focus on the concepts and definitions of dependability, failures, errors, faults and tolerance. First, there is a need to clarify the subtleties involved. Secondly, and possibly more important, is a desire to avoid dependability concepts being reinvented in other research domains such as safety, survivability, trustworthiness, security, critical infrastructure protection, information survivability, and so on [6]. Often the associated research communities do not realize that they are dealing with different facets of the same concept, and are failing to build on existing research advances and insights [6].

This focus on concepts and definitions is also critical for Autonomic Computing. Research and development from many disciplines will be required and, as already mentioned, the successful integration of AI and software engineering, will be particularly important.

In the IBM manifesto for Autonomic Computing [10] its success is linked to the use of open standards, open source code, and open technologies in general. Yet there is also a need for common concepts and indeed common or open definitions for the researchers from the many disciplines that are going to make Autonomic Computing a reality.

On first consideration, dependability and fault tolerance would appear to be specifically aligned to the self-healing facet of Autonomic Computing. Yet any system that is incorrectly or ineffectively configured and/or inefficiently optimized is likely to lead to failures in the future. Similarly, any system that is not adequately protected is vulnerable to malicious faults, be they from hackers or viruses. Thus, essentially all facets of Autonomic Computing are concerned with dependability [36].

Referring again to Randell's fundamental chain:

… ➔failure ➔ fault ➔ error ➔…

and its abstract form:

… ➔event ➔ cause ➔ state ➔…

then each facet of Autonomic Computing (Figure 3) can be considered 'states of undependability' or 'states of dependability' according to how well they are addressed in a system.

<u>States of Undependability</u>
Faulty (unhealthy)
Ill-configured
Sub-optimal
Unprotected

That is, if any of these states exist within a system they are liable to lead to subsequent errors; in turn, that may lead to subsequent faults and on to failure. Autonomic Computing, through self-healing, self-configuring, self-optimization and self-protection, will therefore increase dependability.

4. Towards Autonomicity in Complex Systems

This section takes a look at some exemplar complex areas to highlight the need for autonomicity.

4.1 Telecommunications Systems

As the size and complexity of networks and communications continue to grow, there is a heightened need to develop new techniques capable of achieving a level of service with successful operations upon which users can place even more reliance. Key emerging strategies for meeting this demand is 'autonomic networks' and 'autonomic communications', concepts similar to autonomic computing while specific to the communications field

The Autonomic initiatives are about much more than faults and self-healing, yet it is a critical area to address considering that it has been estimated that companies spend 33% to 50% of their total cost of ownership recovering from or preparing against failures [28]. Also in relation to dependability as presented earlier – all properties within self-management can be related to a fault focus.

The Internet, with its vast infrastructure supporting millions of interconnected computers is perhaps the most significant development. The complexity of networks has grown in various ways [29]. As user demands and expectations become more varied and complex so too do the networks themselves. Data, voice, image, and other information now travels under the control of different protocols through numerous physical devices manufactured and operated by different vendors. It is expected that the trend towards increasing complexity will continue, due to several factors such as the increasing complexity of individual network elements, the need for sophisticated network and communication services and the heterogeneity of connected equipment [30]. The promise of Autonomic Networks, networks that manage themselves, will substantially abate this complexity crisis.

Survivable Network Architectures demonstrate some autonomic behaviour in the physical layer of the telecommunications networks, yet this is just the beginning of the autonomic vision; zero touch, self-sensing, context-aware, dynamic, self-programming and evolvable networks. To create Autonomic Networks will require the co-operation of the industry to develop open standards to evolve from the current network elements (NEs) to autonomic network elements (ANEs). From a Telco's perspective the physical layer tends to be outside their immediate design control as the NEs are supplied by third party vendors.

Telco's offer communications and services across a large variety of technologies. Each technology within the network; SDH (SONET in USA), PDH, ATM, IP and so on, all have their own specific domain technology fault managers. SDH frames may be carrying ATM frames which may be carrying IP and so on. As such at the physical layer Autonomic Networks may resolve their own management issues, but these may have affected the traffic/service they are carrying. This can only be determined at a higher layer.

Essentially due to the complexity the situation has arisen that a large number of uncorrelated alarm event messages may reside on a network at any one time. One estimate concerning BT's UK network was that 95% of all alarm events raised remain uncorrelated, amounting to tens of thousands alarm events being active at any one time. Over time this amounts to a substantial load of data. Another concern that is these problems with root cause analysis are preventing the development of further autonomics particularly in self-healing and with increasing mean-time to human intervention.

Autonomic Networks in themselves will not be an easy goal to achieve, yet the longer term goal of Autonomic Communications is much more than this, having commonality with Ubiquitous and Pervasive Computing, a vision of communications services anytime, anyplace from any device adapting to the users current needs and situation. Effective problem determination in the networks will assist in enabling other autonomics to advance.

The introduction of autonomic principles requires the monitoring of individual system components through sensors and the ability of those components to respond to requests through effectors. Monitoring will typically involve the *correlation* of several related pieces of information. Correlation is important in both self-assessment (self-awareness) and in the assessment of a component's operating environment (environment awareness). This helps in deciding when action is required and what should be done.

By analogy with the human autonomic nervous system, event messages are similar to the electric pulses that travel along nerves. When a fault occurs in an SDH network a series of triggered events are usually reported to the element controller (manager). The behavior of the alarms is often so complex it appears non-deterministic [31] making it very difficult to isolate the true cause of the fault [32]. Yet at this level this is one of the primary goals of Autonomic Networks.

Currently, the skill of the operator is central to identifying faults. So although automation prevents the immediate loss of traffic and preserves the general function of the system (as in the SNA), intervention is necessary to determine and resolve problems that arise. The promise of autonomic networks would bring about a significant reduction in the role of the operator.

IBM concurs with this assessment that root cause analysis in complex systems is key to achieving autonomics. In their white paper 'Autonomic problem determination: A first step towards self-healing computing systems' [33] they state that in effect complexity in problem determination is diluting the effectiveness of computing in the corporate environment. The same can be said for communications and networks. It has been estimated that companies now spend from one third to one half of their total cost of ownership recovering from or preparing against failures [28] While many of these outages, with some estimates at 40%, are caused by operators themselves [34].

One of the major differences that the vision of autonomicity brings to the existing efforts towards advanced automation (often including AI research) is the *situated* aspect – the goal to deal with the problem as local as possible and within the context of the situation. Although the complex telecoms systems have automated fail-over the alarm event messages are passed off to an element manager. The autonomic vision has each component having its own manager.

4.2 Space Flight Systems

Complexity in Space Systems has been well documented [18][19][21]. New paradigms in spacecraft design are leading to radical changes in the way NASA designs spacecraft operations [16]. Increasing constraints on resources, and greater focus on the cost of operations, has led NASA, and other agencies, to use adaptive operations and move towards almost total onboard autonomy in certain classes of mission operations [17],[18].

NASA missions, particularly those to deep space, where manned craft cannot currently being sent, are considering the use of almost wholly autonomous decision-making to overcome the unacceptable time lag between a craft encountering new situations and the round-trip delay (of upwards of 40 (earth) minutes) in obtaining responses and guidance from mission control.

More and more NASA missions will, and *must*, incorporate autonomicity as well as autonomy [19][20], and the Autonomic Computing initiative has been identified by NASA as having potential to contribute to their goals of autonomy and cost reduction in future space exploration missions [18],[19],[20].

ANTS, Autonomous Nano-Technology Swarm, is a mission that will launch sometime between 2020 and 2030 ("any day now" in terms of NASA missions). The mission is viewed as a prototype for how many future unmanned missions will be developed and how future space exploration will exploit autonomous and autonomic behavior.

The mission will involve the launch of 1000 pico-class spacecraft swarm from a stationary factory ship, on which the spacecraft will be assembled. The spacecraft will explore the asteroid belt from close-up, something that cannot be done with conventionally-sized spacecraft.

As much as 60% to 70% of the spacecraft will be lost on first launch as they enter the asteroid belt. The surviving craft will work as a swarm, forming smaller groupings of *worker* craft (each containing a unique instrument for data gathering), a coordinating *ruler*, that will use the data it receives from workers to determine which asteroids are of interest and to issue instructions to the workers and act as a coordinator, and *messenger* craft which will coordinate communications between the swarm and between the swarm and ground control. Communications with earth will be limited to the download of science data and status information, and requests for additional craft to be launched from earth as necessary.

A current project (FAST) is studying advanced technologies for the verification of this incredibly complex mission; the reader is directed to [18][20] for a more detailed exposition of the ANTS mission and the FAST (Formal Approaches to Swarm Technologies) project. Forma approaches to verification of such complex autonomic systems are essential, as all possible behavior cannot possibly be determined in advance, and no *a priori* testing plan is likely to be realistic.

5. Discussion

There is a need to establish standards and mechanisms for autonomic computing to work. For instance it is possible to develop a self-healing tool with a control loop that constantly monitors the processes running on your laptop [35]. If any should hang, the autonomic tool can restart that process. Yet there is no means to inform the process where to restart – effectively it's a process being started from fresh with any previous state lost unless the process' application itself handles this. There is a need for standards for autonomic signals and communications to take place not only at this level – autonomic manager to processes running on the managed component but also autonomic manager to autonomic manager. Allowing standard 'autonomic signal' routes into processes would raise security issues – yet this will need to be part of the self-protection autonomic property.

Since this implies all processes effectively need to be designed with autonomicity and self-managing capabilities in mind – not only from within but taking direction from the external environment – this not only raises issues of standards to achieve this but raises questions do the current design and development approaches meet the needs for developing autonomicity and handle human error due to complexity. The realisation of self-managing systems, which will still be complex to design, may only move the human error aspect from the administrator (who had been manually managing the running systems) to the designer.

The telecommunications domain was discussed as an exemplar as its alternative evolution may consider it as much further down this path then the computer industry – i.e. its systems have a management layer, with standards allowing heterogeneous elements to communicate management information. Consider how often your phone goes down compared to you PC or internet connection. Yet the design of the management layer has created a complex system in itself where it has been claimed 95% of the event messages under fault conditions cannot be automatically correlated and this has created a bottleneck for further advanced automation. It is essential that the emerging Autonomic research community find a way forward to deal with this hard problem of root cause analysis from the start and avoids this situation.

The NASA example illustrates a complex system that cannot be managed from Earth due to bandwidth limits and time delays. Moreover, it is a complex system where decisions need to be made with real time constraints. Even without bandwidth issues, the system would likely be too complex to be managed in real time by human beings. Fully autonomous behavior is realistically the only alternative. But, in order for this to be successful, the mission *must* include the autonomic properties of self-healing, self-protecting, self-configuring and self-optimizing. In short: the mission must be self-managing.

6. Conclusion

Autonomic computing is an emerging holistic approach to computer system development that aims to cope with complexity and bring a new level of automation and dependability to systems through self-healing, self-optimizing, self-configuring and self-protection functions.

To illustrate that autonomicity may assist with coping with complexity, examples from research in telecommunications and space systems where discussed.

While Autonomic Computing may not be a panacea for complex computer system, it clearly does have a role to play in overcoming complexity, and offers a promising antidote to some of the problems of complex systems.

Open standards and technologies are required for Autonomic Computing to reach its goals. The challenges of addressing these issues must be taken up by the wider computing community. A remit needs to be established to encourage interaction and research among researchers and developers in industry, government and academia in determining standards, techniques, development processes and mechanisms that can be exploited in creating self-managing systems.

Acknowledgements

The development of this paper was supported at University of Ulster by the Centre for Software Process Technologies (CSPT), funded by Invest NI through the Centres of Excellence Programme, under the EU Peace II initiative.

Part of this work has been supported by the NASA Office of Systems and Mission Assurance (OSMA) through its Software Assurance Research Program (SARP) project, Formal Approaches to Swarm Technologies (FAST), and by NASA Goddard Space Flight Center, Software Engineering Laboratory (Code 581).

References

[1] J. Gray, “Dependability in the Internet Era”, http://research.microsoft.com/~gray/talks/InternetAvailability.ppt.

[2] P. Horn, Invited Talk to the National Academy of Engineering at Harvard University, March 8, 2001

[3] D. Patterson, Recovery-Oriented Computing, Keynote, at High Performance Transaction Systems Workshop (HPTS), October 2001

[4] J. Gray, “What Next? A dozen remaining IT problems” Turing Award Lecture, FCRC, May 1999

[5] Per Hour Downtime Costs, Contingency Planning Research and Internetweek (4/3/2000)

[6] B. Randell, “Turing Memorial Lecture – Facing Up to Faults”, Comp. J. 43(2), pp 95-106, 2000.

[7] A. Avizienis, J.-C. Laprie, B. Randell, “Fundamental Concepts of Dependability”, UCLA CSD Report #010028, 2000.

[8] J.C. Laprie, “Dependability: basic concepts and terminology-in English, French, German, Italian and Japanese”, In Dependable Computing and Fault Tolerance, p.265, Springer-Verlag, Vienna, 1992.

[9] J.C. Laprie, “Dependable computing: concepts, limits, challenges”. In Proceedings 25th IEEE International Symposium on Fault-Tolerant Computing –Special Issue, Pasadena, CA, pp42-54, 1995.

[10] P. Horn, "Autonomic computing: IBM perspective on the state of information technology", IBM T.J. Watson Labs, NY, 15th October 2001, Presented at AGENDA 2001, Scotsdale, (http://www.research.ibm.com/autonomic/), 2001

[11] E. Mainsah, "Autonomic computing: the next era of computing", IEE Electronics Communication Engineering Journal, Vol. 14, No. 1 (Feb), pp2-3, 2002

[12] A. Wolfe, "IBM sets its sights on 'Autonomic Computing'", IEEE Spectrum, Jan., p18-19, 2002

[13] L.D. Paulson, "IBM Begins Autonomic Computing Project", IEEE Computer, Feb., p25, 2002.

[14] R. Sterritt, “Towards Autonomic Computing: Effective Event Management,” Proceedings of 27th Annual IEEE/NASA Software Engineering Workshop (SEW), Maryland, USA, December 3-5 2002, IEEE Computer Society Press, pp 40-47.

[15] A. Avižienis, J-C Laprie, B Randell, C Landwehr, “Basic Concepts and Taxonomy of Dependable and Secure Computing”, IEEE Transactions on Dependable & Secure Computing, 1(1), Jan.-Mar 2004.

[16] M.A. Swartwout, “Engineering Data Summaries for Space Missions,” SSDL, 1998.

[17] J. Wyatt, R. Sherwood, M. Sue, J. Szijjarto, “Flight Validation of On-Demand Operations: The Deep

Space One Beacon Monitor Operations Experiment," 5th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS '99), ESTEC, Noordwijk, The Netherlands, 1-3 June 1999.

[18] W. Truszkowski, M. Hinchey, J. Rash and C. Rouff, "NASA's Swarm Missions: The Challenge of Building Autonomous Software," IEEE IT Professional mag., September/October 2004, pp 51-56.

[19] W. Truszkowski, M. Hinchey, C. Rouff and J. Rash, "Autonomous and Autonomic Systems: A Paradigm for Future Space Exploration Missions," *submitted for publication*.

[20] W. Truszkowski, J. Rash, C. Rouff and M. Hinchey, "Asteroid Exploration with Autonomic Systems," Proceedings of IEEE Workshop on the Engineering of Autonomic Systems (EASe 2004) at the 11th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS 2004), Brno, Czech Republic, 24-27 May 2004, pp 484-490.

[21] R. Sterritt, "Pulse Monitoring: Extending the Health-check for the Autonomic GRID," Proceedings of IEEE Workshop on Autonomic Computing Principles and Architectures (AUCOPA 2003) at INDIN 2003, Banff, Alberta, Canada, 22-23 August 2003, pp 433-440.

[22] The Autonomic Computing Workshop, 5th Int. Workshop on Active Middleware Services (AMS 2003), Seattle, WA, Proceedings IEEE Computer Society, pp198, 25th June 2003.

[23] IJCAI Workshop, "AI and Autonomic Computing: Developing a Research Agenda for Self-Managing Computer Systems, Acapulco, Mexico, August 10, 2003, http://www.research.ibm.com/ACworkshop

[24] Workshop on Autonomic Computing Principles and Architectures (AUCOPA' 2003), at INDIN 2003 - First IEEE Conference on Industrial Informatics, Banff Canada, August 2003.

[25] 1st International Workshop on Autonomic Computing Systems at 14th International Conference on Database and Expert Systems Applications (DEXA'2003). Prague, Czech Republic September 1-5, 2003.

[26] Autonomic Applications Workshop, at Int. Conf. High Performance Computing (HiPC 2003), Taj Krishna, Hyberabad, India, 17th December 2003

[27] IEEE ECBS Workshop on Engineering of Autonomic Systems (EASe 2004), Brno, May 2004.

[28] Patterson, D.A., Brown, A., Broadwell, P., Candea, G. , Chen, M., Cutler, J., Enriquez, P., Fox, A., Kiciman, E., Merzbacher, M., Oppenhiemer, D., Sastry, N., Tetzlaff, W., Traupman, J., Treuhaft, N., 2002, Recovery-Oriented Computing (ROC): Motivation, Definition, Techniques, and Case Studies, U.C. Berkeley Computer Science Technical Report, UCB//CSD-02-1175, University of California, Berkeley, March 15.

[29] Oates, T., 1995, Fault identification in computer networks: A review and a new approach. Technical Report 95-113, University of Massachusetts at Amherst, Computer Science Department.

[30] Cheikhrouhou M., Conti P., Labetoulle J., Marcus K., 1999, Intelligent Agents for Network Management: Fault Detection Experiment. In Sixth IFIP/IEEE International Symposium on Integrated Network Management, Boston, USA, May.

[31] Bouloutas, A. T., Calo, S., Finkel, A., 1994, Alarm correlation and fault identification in communication networks, IEEE Trans. on Comms., 42(2), pp 523-533.

[32] Klemettinen, M, 1999, A knowledge discovery methodology for telecommunication network alarm databases, Ph.D. Thesis, University of Helsinki, Finland.

[33] IBM, 2003, Autonomic problem determination: A first step towards self-healing computing systems, White Paper, October

[34] Patterson, D., 2002, Availability and Maintainability Performance: New Focus for a New Century, USENIX Conference on File and Storage Technologies (FAST '02), Keynote Address, Monterey, CA January 29

[35] Sterritt R, Chung S, (May 2004) *"Personal Autonomic Computing Self-Healing Tool"*, Proceedings of IEEE Workshop on the Engineering of Autonomic Systems (EASe 2004) at 11th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS 2004) , Brno, Czech Republic, 24-27 May, Pages 513-520

[36] Sterritt R, Bustard DW, (Apr 2003) "Autonomic Computing-a Means of Achieving Dependability?", Proceedings of IEEE International Conference on the Engineering of Computer Based Systems (ECBS'03), Huntsville, Alabama, USA, April 7-11, IEEE CS Press, Pages 247-251